Multicast Service Discovery Profiles for Deployment within Dynamic Edge Networks

Joseph Macker, Justin Dean
Naval Research Lab
Washington DC, USA
joseph.macker@nrl.navy.mil
jdean@itd.nrl.navy.mil

Ian Taylor, Andrew Harrison
School of Computer Science
Cardiff University, Cardiff, UK
Ian.J.Taylor@cs.cardiff.ac.uk
and A.B.Harrison@cs.cardiff.ac.uk

***Abstract*—Deploying survivable group-oriented applications and enterprise services across network infrastructures incorporating the use of mobile ad hoc edge networks is of interest in military communication scenarios. This requires a more flexible approach to service discovery than conventional solutions typically provide. In this paper, we discuss our service discovery design extensions leveraging improved multicast capabilities in multi-hop, wireless networks. The design also supports multiple discovery profiles, through the use of flexible timing parameters, caching and operating modes. We also provide an overview of our working software prototype and methodology used in examining scenario-based system performance. Our experiments investigate various service provider and consumer distributions across a set of mobile ad-hoc network scenarios. Our experimental scenarios involve node mobility and varying degrees of temporal connectivity. We illustrate the differences in success rates, time delays and network overhead of several discovery profiles. Results indicate that a one-size-fits all profile is not practical and that a flexible means of deploying specific discovery profiles for different applications is needed to optimize performance.**

I. Introduction

There has been extensive research work related to dynamic wireless networking, including the often cited case of mobile ad hoc networks (MANETs) [1]–[3]. Significant DoD, academic, standards, and industry focus has been placed on the development and testing of routing protocol variants for MANET environments. Much of this work focuses on the development of unicast routing capabilities and to a lesser extent on multicast routing and forwarding [4]–[6]. There are now a myriad of MANET routing solutions to choose from and applied deployment experience has improved (e.g., operational community networks [7]). However, far less research has been focused on upper layer distributed network services and collaborative applications operating within such networks. Here we address a related technology gap by examining enhanced network service discovery methods for use within tactical edge networks. Our goal is to develop and identify decentralized and mobility tolerant service discovery solutions that can operate effectively within tactical edge networks. We are also interested in supporting interconnection between fixed and mobile network architectures as depicted in Figure 1.

Use cases cover a broad range of possibilities, not only in our focus area for tactical edge military applications, but also in related areas, such as medical emergency scenarios and

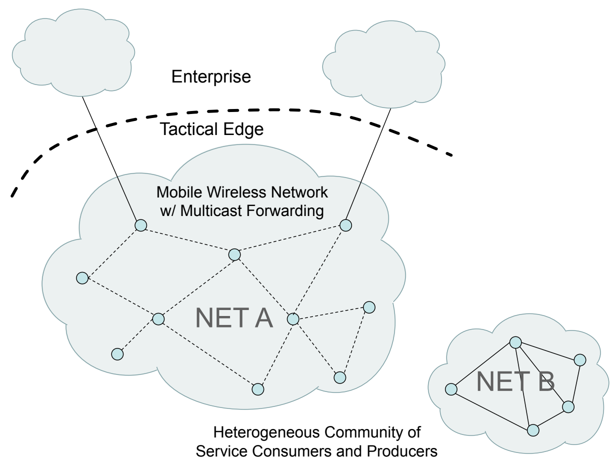

Fig. 1. Tactical Edge Architecture showing a Mobile Multicast Edge Network

disaster response. Mobile network scenarios range in infrastructure types from highly autonomous MANET operations to the hybrid use of unidirectional satellite links and cellular systems. A general design challenge is to operate a distributed network service layer above a physical network architecture that is subject to dynamics, as shown in Figure 2. Since service discovery in mobile edge networks require additional design considerations [8], its mechanism benefits from a variety of operational modalities (unicast, multicast, reactive, i.e. solicited service advertisement publishing, proactive, i.e. unsolicited service advertisement publishing) and configuration flexibility (e.g. retransmission timers, data cache settings).

In the application space, there are a number of motivational examples illustrating the need for service discovery within MANET networks. Dynamic situational awareness and collaboration, through effective discovery and sharing of your operational environment and related services, is a key motivational concept. As an example, mobile platforms and assets may dynamically publish local services, discover and maintain other relevant mission and application services, and gain access to other collaborative services. Collaborative applications may include: chat, VoIP, command and control for providing orders, alarms, reports; data services, such as in-theatre video, real-time sensors (e.g. weather, sensor data, maps), access to caches of historical data, logistical support; and composable application and workflow support that benefits

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
NOV 2010		**00-00-2010 to 00-00-2010**

4. TITLE AND SUBTITLE **Multicast Service Discovery Profiles for Deployment within Dynamic Edge Networks**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Research Laboratory,Washington,DC,20375**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Presented at MILCOM 2010 Military Communications Conference, Oct 31, Nov 3, 2010, San Jose, CA

14. ABSTRACT
Deploying survivable group-oriented applications and enterprise services across network infrastructures incorporating the use of mobile ad hoc edge networks is of interest in military communication scenarios. This requires a more flexible approach to service discovery than conventional solutions typically provide. In this paper, we discuss our service discovery design extensions leveraging improved multicast capabilities in multi-hop, wireless networks. The design also supports multiple discovery profiles, through the use of flexible timing parameters caching and operating modes. We also provide an overview of our working software prototype and methodology used in examining scenario-based system performance. Our experiments investigate various service provider and consumer distributions across a set of mobile ad-hoc network scenarios. Our experimental scenarios involve node mobility and varying degrees of temporal connectivity. We illustrate the differences in success rates, time delays and network overhead of several discovery profiles. Results indicate that a one-size-fits all profile is not practical and that a flexible means of deploying specific discovery profiles for different applications is needed to optimize performance.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**6**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

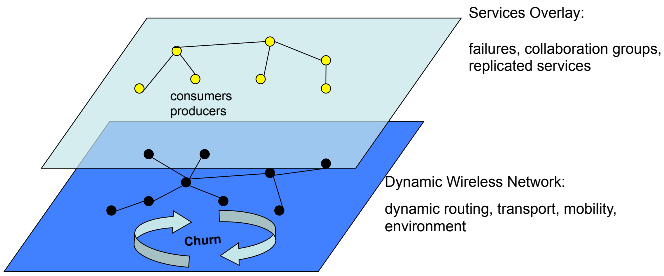

Fig. 2. Service Overlay

from on-the-fly collection and management of services and data. The middleware used to exchange messages may also vary, depending on the nature of the application (e.g. XMPP [9] for collaborative systems and WS-Notification [10] for publish/subscribe systems.)

In this paper, we consider two example notional use cases that require different combinations of service providers and consumers:

- **Hierarchical Command and Control, Single Provider Service** – involves multiple consumers and typically one provider of information, rising to several depending on the specific application. Here, notional force order updates (data) are issued by a single point and consumed by many participants. Consumers may need to discover and maintain access to the command and control service or service components, as the network topology changes. This service type can also represent other in-theatre dynamic services such as real-time video feeds making them more useful in distributed environments.
- **Peer Group Situational Awareness and Collaborative Services** – an example of multiple consumers and multiple providers. The collaboration could involve service components: database exchanges, cataloging of service locations (DA), tracking information, real-time data streams, sensor data sharing, or even locations of media files. For our experiment presented in Section IV, we focus on showing one consumer interacting with multiple providers within a situational awareness use case, and observe how discovery profiles impact the results.

These two examples provide two diverse cases and are not intended to be all inclusive but indicate examples where reactive or proactive discovery modes might be appropriate. For example, in cases where service consumers far outweigh service providers (e.g. our command and control use case), we show that proactive or mixed mode discovery offers a more robust approach. When there is more availability of service providers (e.g. our situational awareness use case), we show that a reactive approach performs equally as well with potentially better response times. Collectively, these experiments demonstrate that a single discovery approach with fixed parameters is not effective across all possible combinations of network mobility and usage scenario. We also vary as independent variables the degree of connectivity and mobility of these two use cases to illustrate how parameter choice and discovery mode influences performance.

The rest of the paper is organized as follows. The next section provides an overview of the enhancements on existing discovery solutions that we focus on in this research. Section III describes the design and implementation of the resulting discovery system that encompasses these enhancements. Section IV shows results for the two deployment scenarios, performed for multiple levels of mobility and connectivity. In Section V we summarize our findings and state future work to be undertaken in the area.

II. Discovery Advancements over Previous Work

A combined solution of MANET-appropriate multicast forwarding and multicast-enhanced service discovery can improve wireless resource usage, system survivability, and mobile collaborative interaction for a myriad of applications. Our work has resulted in the service discovery software toolkit shown in Figure 3 that can deployed across standard IP multicast capable networks. We call our prototype the Independent Network Discovery Interface (INDI[1]). INDI extensions enhance the state of the art of other widely deployed discovery systems (e.g. multicast Domain Name Service with Service Discovery (mDNS-SD or Apple bonjour) [11], [12] and the Simple Service Discovery Protocol (SSDP) in Microsoft's Universal Plug n' Play (UPnP) standard [13]). The multicast design focus of these existing protocols has been primarily at the local network interface level and we provide additional multiple hop multicast and dynamic operation capabilities. The INDI prototype design also borrows some concepts from another existing standardized service discovery solution (Service Location Protocol V2 (SLPv2) [14], [15]). We also extend existing designs by developing a variety of reactive and proactive operational modes and offering finer parameter control to provided a flexible working prototype to aid performance investigations in under scenario-specific conditions.

Due to the mobile nature of edge networks and the desire for increased robustness through decentralization, there is a strong system design and performance link between improved multicast capability and related service discovery or presence capabilities. In this regard, dynamic multicast forwarding capability is a key enabler for more effective mobile edge network services. Although our approach is independent of the multicast routing or forwarding protocol, our experiments use the Simplified Multicast Forwarding (SMF) protocol [2] as a means to provide dynamic group packet delivery. This solution provides an administratively scoped multi-hop capability versus the more typical single network hop IP link-local multicast capability used by many deployed service discovery approaches. Although we decouple the specifics of the underlying network routing and transport approach from INDI, there is a strong relationship between mobile proactive notification, querying, opportunistic caching and multicast forwarding.

These same types of optimized flooding algorithms are also available in many ad hoc routing control planes and existing research has demonstrated the ability to encapsulate service

[1] INDI means data, information, indication or pointer in Latin
[2] SMF Draft is at https://tools.ietf.org/html/draft-ietf-manet-smf

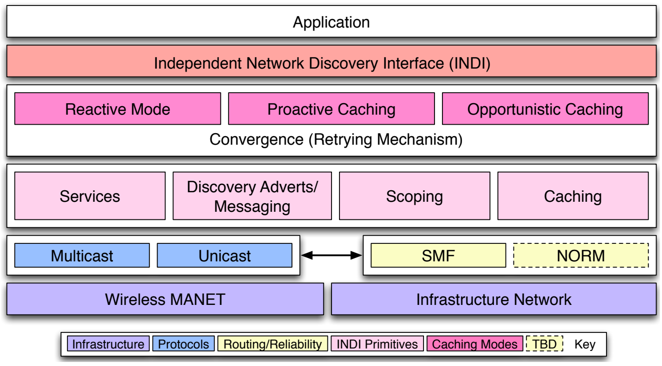

Fig. 3. INDI Architecture

discovery messaging within the control plane mechanism [16]. We do not suggest a particular approach to the message forwarding here, as there are many valid options and INDI supports standard IP multicast traffic but can also be adapted for middleware encapsulation. Rather, we focus on examining the use of such techniques, whether encapsulated or not, to enable the potential for more robust distributed querying, response, and proactive notification.

III. INDI Overview

The present INDI implementation is toolkit for network service discovery and notification. As illustrated in Figure 3 INDI provides several layers focusing on different levels of the discovery stack. Our motivation for developing INDI is to achieve a modular research platform for investigating lightweight, scalable and flexible service discovery paradigms within a variety of network architectures, focusing primarily on its use in dynamic, mobile environments. To this end, INDI supports profiles that can contain one of three modes of discovery in order to address different network dynamics: Reactive, Proactive and Opportunistic Caching.

To support these three modes of operation, INDI leverages some design elements from SLPv2 [14]. INDI uses SLP's core roles: User Agent (UA) — a service consumer; Service Agent (SA) — a service provider; and Directory Agent (DA) — a cache/proxy for service advertisements. In the INDI design, each role (UA, SA, DA) is configured using one of the three operational modes. Each mode is constrained by a few important parameters: *timeouts* – a list of exponentially increasing wait intervals; *required results* – the number of successful connects the UA requires; and *maximum retries* – the number of retries to attempt per request. These parameters define fault tolerance behavior beyond any provided by the underlying network stack, which provide resilience to the dynamically changing connectivity and mobility patterns.

Shown in the lower part of Figure 3 is a pluggable interface for supporting multiple unicast and multicast protocol stacks. In this paper we use the SMF protocol to provide multicast forwarding. No additional reliable transport protocol was used here but we are considering protocols such as NORM [17] for potential reliable multicast and unicast improvements.

A. Reactive Mode

The reactive mode provides a more conventional means of service discovery. A consumer dispatches a service request

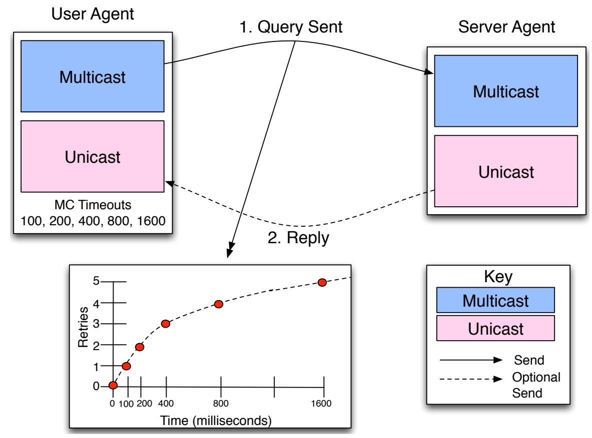

Fig. 4. Reactive Mode

using a query (containing attributes or a template for matching purposes) and providers respond with a service advert, which contains service endpoint identification and any additional metadata related to that service. In brokered systems, the service contact might be a centralized registry which may differ from the service endpoint location. Many discovery subsystems e.g. Jxta [18] and Jini [19], use a combination of unicast to contact directories and multicast to search multicast groups for peers that have services on the network. For this paper, we focus on the MANET case so there is a greater interest in the decentralized multicast discovery mechanisms.

Figure 4 illustrates this mode for a MANET. An INDI UA performs a query in a standard fashion for a decentralized network, i.e., using multicast, and then performs a retransmission backoff algorithm defined by the *timeouts* parameter. For these experiments the initial wait time is set to 100 milliseconds (set to around twice a typical response delay in our test environment) and exponential retransmission backoff is performed beyond this using a *maximum retries* setting of 5. This mechanism is analogous to the convergence algorithm used by SLPv2 and supports the inclusion of "previous responders" in the query, allowing providers to know whether a consumer has already received a response from them. In addition, INDI supports a *retry* field representing the times the consumer has attempted a request within a single retransmission session. This allows providers to not re-send a response to a particular retry even if the consumer never received their response was therefore unable to insert them as a previous responder.

In the experiments, a service discovery event is not restricted to the process of finding a matching service advertisement. The additional step of actually attempting to connect the service provider described in the service advert is performed. The outcome of the connect process encompasses a total discovery process from discovery through the connect stage. This feature is important in dynamic environments where service adverts are more likely to become stale due to dynamics.

Therefore, if a matching service advert is received, a connect thread is initiated. The connection process in our experimental model uses the same exponential backoff algorithm as the querying process. It determines how many providers it can connect to at each timeout and attempts as many as it can. In our simulations *connection interval* is set to 100 milliseconds.

Hence, if the amount of time allocated to the connection thread is 400 milliseconds, then up to four providers can be connected to in that time period. Of course the actual number of providers to attempt is further bounded by the actual number of appropriate service adverts known. The results described in Section IV show successful retries that result in an actual service connection. Corresponding discovery rates are also shown for comparison.

B. Proactive Discovery

In proactive discovery the providers, SAs and DAs, perform a service push by periodically sending out their service adverts using multicast. These adverts are delivered to all consumers subscribed to the service discovery multicast group, as illustrated in Figure 5. The effectiveness of the service push periodicity is dependent on the dynamic nature of the network; more frequent pushes may deal with increased dynamics at the expense of increased overhead. INDI UAs operating in the proactive mode use an internal cache, which caches service adverts received from producers.

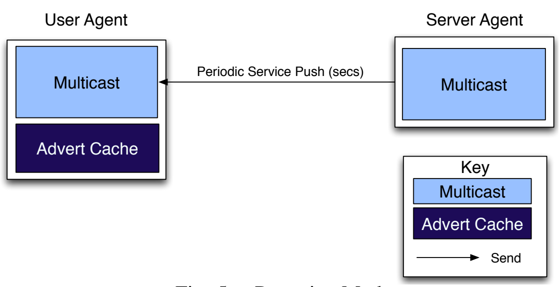

Fig. 5. Proactive Mode

When a UA invokes a service request, it first checks its local cache for valid matching service adverts. If matching services are found it attempts to connect to a service (discovery time 0), otherwise it sleeps and waits for the next interval before checking again. This continues until the maximum number of retries are exhausted or enough service connections are successful. The primary difference between reactive and proactive mode in terms of consumer behavior is that consumers do not issue service requests during the convergence process in proactive mode, instead they wait to receive adverts from providers.

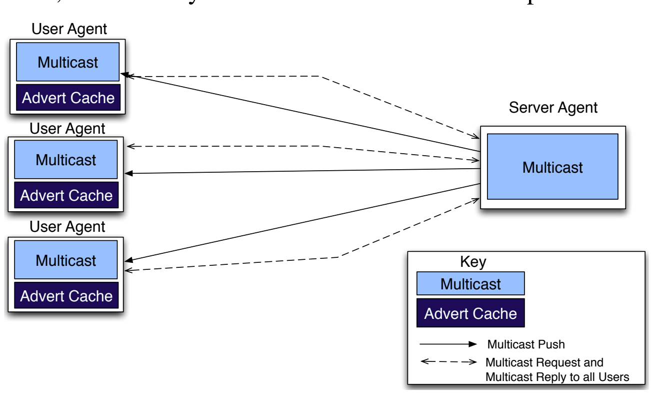

Fig. 6. Opportunistic Caching Mode

C. Opportunistic Caching or Hybrid Discovery

The distributed opportunistic caching scheme is based on the concept that if a consumer queries for a service, other consumers may be interested in that service and can opportunistically cache them for future use. Opportunistic caching UAs will automatically cache all adverts overheard on the network. When an opportunistic caching UA requires a service, it will first check its local cache for valid matching service adverts. Like proactive mode, if there are matching adverts available, it will attempt to connect to the appropriate end point. Otherwise, it will revert to reactive mode and begin the reactive multicast service request process. The main difference between opportunistic caching and reactive is that using opportunistic caching, SAs and DAs will reply with multicast service adverts, as opposed to a unicast adverts. This allows any other opportunistic caching UA to potentially overhear and cache that information for possible later recall.

IV. Simulations for Application Scenarios

To capture the effects of two scenarios identified in Section I, we chose a testbed of 50 mobile nodes and performed simulations for both scenarios with: 1 provider and 25 consumers (Command and Control Scenario) and 25 providers and 1 consumer (Situational Awareness Scenario). In the simulations, all service providers are equal, nodes never host both consumers and providers (for modeling purposes to force non-local discovery) and the selection of services and consumers is consistent across scenarios regardless of motion scenarios and service discovery settings. We tested the following INDI discovery profiles:

- **Opp Cache:** opportunistic caching mode where each service expires after 9 seconds.
- **Opportunistic Cache 54 Sec:** opportunistic caching with each service lifetime set to 54 seconds.
- **Proactive Cache 3 Sec:** proactive mode where each node multicasts their service onto the network every 3 seconds.
- **Proactive Cache 18 Sec:** proactive mode where the adverts are propagated every 18 seconds.
- **Reactive:** Reactive mode

For Proactive, the service lifetime is set to three times the periodicity of the service pushes onto the network. Therefore, the opportunistic caching and proactive schemes employ the same service lifetime for comparison; that is, 9 and 54 seconds.

Aside from the two core scenarios and modes, we also wanted to explore how different mobility and connectivity profiles would impact the results of the experiment. For mobility, we chose a modified Random Walk motion model which independently adjusts speed and directional heading (+/- 1m/s and +/- 15 degrees), with limited scope (0-5 m/s), for each node at every selected time interval (1 second). This approach maintains a more uniform distribution of nodes within in a bounded grid space over often-used random motion techniques like the random vector waypoint model [20], [21]. For "connectivity", we used a normalized measure of network connectivity to categorize different random mobility scenarios. A time snapshot of the mobile network was taken at regular intervals, represented by undirected graph G. We defined mobile network connectivity (N_c) metric as the average expected fraction of the network reachable from a randomly selected node, measured across all topological time intervals, as follows:

$$N_c = (\sum_{t=1}^{T}(\sum_{n=1}^{N}|C_n|/N^2)/T \qquad (1)$$

where $|C_n|$ is the order of the connected component in G containing node n, T is the number of uniform time intervals within the mobility scenario (300 in our case), and N is the number of nodes or vertices, V, within G. A N_c value of 1.0 (100%) is a fully connected network at all times. Lesser values indicate partial network connectivity for at least some time portion of the simulation. Using the N_c metric and we generated ten separate random mobility scenarios with resulting connectivity coefficients of 60, 70, 80, 90, and 100, resulting in 50 mobility profiles per scenario. Mobility profiles with higher connectivity coefficients were generated using smaller mobility areas resulting in higher network densities than those with lower coefficients. This allowed settings for Random Walk node mobility generation to be constant across all 50 mobility profiles, resulting in similar average link change rates (1.5 to 2.0 percent per second) for each profile.

INDI used the AgentJ [22], [23] toolkit to execute the unmodified Java INDI code within the ns2 environment [24] (i.e. we ran *actual code* rather than discretizing our approach).

To generate the scenario files and analyze the results, we developed a software analysis package which defines the test environments, combines the appropriate mobility scripts, populates the service agents, and establishes other parameter settings. Consumer discovery query events were generated randomly using a series of event times according to a Poisson distribution with a mean interval of 6 seconds.

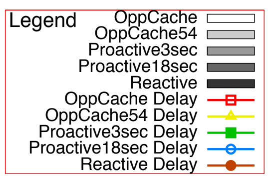

Fig. 7. Legend for 8, 9 & 10

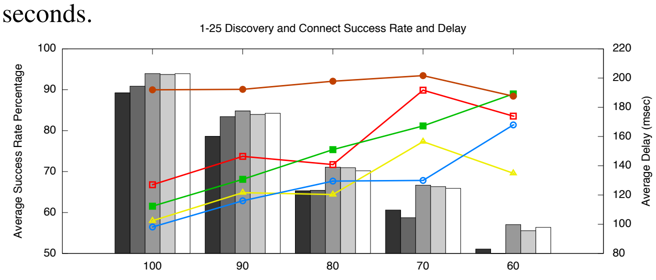

Fig. 8. Discovery/Connect Success with Delay (1 provider, 25 consumers)

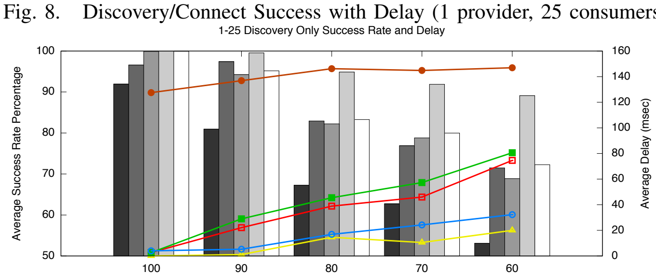

Fig. 9. Discovery Success with Delay (1 provider, 25 consumers)

Figure 8 shows the coupled "successful discovery and connect" results for the 1 provider and 25 consumer simulation scenario (1-25, Command And Control). It can be seen in this case that generally the proactive-based protocols outperform reactive for discovery, both in terms of success rate and overall responsiveness (delay). However, in Figure 9 which shows only the discovery rates, it can be seen that the underlying multicast algorithms perform far better across the board. The decrease in success rates between Figures 8 and 9 is far more pronounced for the proactive-based protocols. In the case of long expiration intervals, valid discovered service advert endpoints have a higher failure rate due to network dynamics, resulting in a larger difference between discovery and success rates. For discovery and connect, the reactive protocol has the lowest success rates, while the proactive protocol with the longer (18 second) expiration interval comes in a close second. Both the opportunistic caching using a 9 second lifetime and proactive using 3 second intervals perform well with respect to success rate in our scenarios, showing their ability to respond well to the dynamics of the network. Although, the proactive protocol performs well across our test scenarios, opportunistic caching proves a competitive combination when considering both success rate and delay metrics.

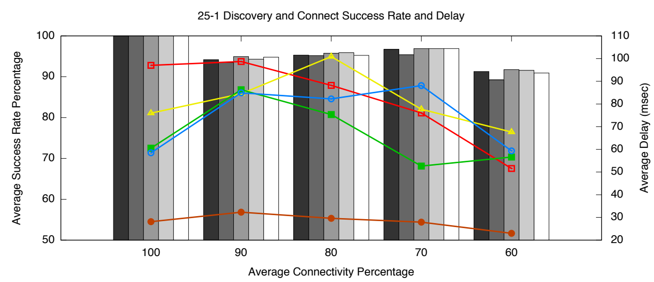

Fig. 10. Discovery/Connect Success with Delay (25 providers, 1 consumer)

For the 25 providers and 1 consumer (25-1, Collaborative) scenario, shown in Figure 10, there is little significant difference between the discovery schemes. This is somewhat intuitive since we are achieving additional robustness through an abstracted model of service replication. The provider locations are so distributed that the probability of finding one of them in a network containing 25 is high even in the more fragmented cases. However, the reactive protocol here offers the best balance between success rate and responsiveness with delay values almost half that of the other protocols. However, when message overhead (combined multicast and unicast) is considered:

Connectivity	60	70	80	90	100
Reactive Mode	649	796	952	1.1K	1.2K
Proactive Mode	3K	3K	3K	3K	3K
OppCach Mode	295	386	535	599	717

then opportunistic caching proves to be an excellent compromise between proactive and reactive protocols for this scenario. The results from Figure 10 show that not only does it provide comparable performance, but it also generates far less overhead than the other two protocols. Opportunistic caching generates around 40% less messages than a reactive protocol and 76% less messages than the competing proactive protocol.

With respect to providing further fault tolerance to the underling multicast algorithms through the use of INDI "retries" in the convergence algorithm, we can see the following gains:

Type	Retries	No Retries
1-25	67.23%	54.9%
25-1	95.3%	93.0%

For low numbers of distributed providers, it can be clearly seen that there is a significant gain in successful discovery rates by using retries. For, 1-25 there is a 13% performance gain and for 25-1 there is a 2% gain. Our findings also show more significant gains are realized when applying retries in scenarios of lower connectivity degree. In these cases, the network forwarding is less robust and the network mobility may also become a positive factor in discovery success of servers that may be initially unreachable.

V. Conclusion and Future Work

Supporting a distributed network service paradigm is vital to hiding the physical network structure and configuration from end users. Such a capability is essential in dynamic, self-organizing wireless networks in which network nodes may join, leave, fail, or move within a given architecture. Also addresses may change and services may migrate. This paper has discussed extensions to existing service discovery protocol frameworks that we feel address many issues for effective operation within wireless, multi-hop mobile network architectures. We have shown that discovery flexibility, in terms of operating modes and parameter settings, is vital for appropriately operating service discovery in different deployment profiles, degrees of connectivity, and mobility patterns. Particularly, early findings indicate that proactive profiles are a better fit when demand far exceeds supply and that an opportunistic caching scheme proves both efficient in terms of success rate and message overhead for scenarios where supply far exceeds demand.

Beyond our initial design and findings there is much that remains to be explored. One area involves modeling additional system heterogeneity in terms of service types and network resources beyond our present models. It is also presently not well understood if or how reliable transport protocols, directory agents, or service proxying techniques provide useful performance gains in MANET type service discovery architectures. Through future studies we hope to document more about the architectural tradeoffs and design options to aid future system designs. Finally, we are also migrating INDI to support interfaces to other service discovery frameworks (e.g., WS-Notify, mDNS-SD) to support interoperable system capabilities.

Acknowledgment

INDI has been developed by the Networks and Communication Systems Branch of the Information Technology Division (ITD) at NRL. On-going modifications to the core system are being funded by the Office of Naval Research (ONR) Service Oriented Networks Operating in MANET (SONOMA) project. The authors would like to thank Brian Adamson for his advice and for the use of his SMF implementation.

References

[1] S. Basagni, M. Conti, S. Giordano, and I. Stojmenović, *Mobile Ad Hoc Networking: Edited by Stefano Basagni...[et Al.]*. IEEE, 2004.

[2] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *MobiCom '98: Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*. New York, NY, USA: ACM, 1998, pp. 85–97.

[3] C. Perkins, *Ad Hoc Networking*. Addison-Wesley Professional, 2000.

[4] S. Lee, M. Gerla, and C. Chiang, "On-demand multicast routing protocol," in *proceedings of IEEE WCNC*, vol. 99. Citeseer, 1999, pp. 1298–1302.

[5] T. Camp, J. Boleng, and V. Davies, "Wireless communications and mobile computing (wcmc): Special issue on mobile ad hoc networking: Research, trends and applications; a survey of mobility models for ad hoc network research," vol. 2, no. 5, pp. 483–502, 2002.

[6] J. Macker, W. Chao, and J. Dean, "Simplified multicast forwarding in mobile ad hoc networks," Naval Research Lab Washington DC Information Technology Div, Tech. Rep., 2004.

[7] "Funkfeuer Free Net." [Online]. Available: http://www.funkfeuer.at

[8] M. Corson, J. Macker, and G. Cirincione, "Internet-based mobile ad hoc networking," *IEEE Internet Computing*, vol. 3, no. 4, pp. 63–70, 1999.

[9] "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence." [Online]. Available: http://tools.ietf.org/html/rfc3921

[10] M. Humphrey *et al.*, "State and Events for Web Services: A Comparison of Five WS-Resource Framework and WS-Notification Implementations," in *14th IEEE International Symposium on High Performance Distributed Computing (HPDC-14)*, 2005.

[11] M. Giordano, "DNS-Based discovery system in service oriented programming," *Lecture notes in computer science*, vol. 3470, p. 840, 2005.

[12] D. Steinberg and S. Cheshire, *Zero Configuration Networking: The Definitive Guide*. O'Reilly Media, Inc., 2005. [Online]. Available: http://portal.acm.org/ft_gateway.cfm?id=1201080&type=safari&coll=GUIDE&dl=GUIDE&CFID=67573037&CFTOKEN=43706727

[13] S. Helal, "Standards for service discovery and delivery," *IEEE pervasive computing*, pp. 95–100, 2002.

[14] "The Service Location Protocol." [Online]. Available: http://www.ietf.org/rfc/rfc2608.txt

[15] E. Guttman, "Service location protocol: Automatic discovery of IP network services," *IEEE Internet Computing*, 1999.

[16] L. Li and L. Lamont, "A lightweight service discovery mechanism for mobile ad hoc pervasive environment using cross-layer design," in *Proc. 3rd IEEE International Conference on Pervasive Computing and communications Workshops, March2005*. Citeseer, pp. 55–59.

[17] B. Adamson, C. Bormann, M. Handley, and J. Macker, "Multicast Negative-Acknowledgment (NACK) Building Blocks," RFC 5401 (Proposed Standard), Internet Engineering Task Force, Nov. 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5401.txt

[18] D. Brookshier, D. Govoni, N. Krishnan, and J. C. Soto, *JXTA: Java P2P Programming*. Sams Publishing, Indianapolis, 2002.

[19] "The Jini Web site." [Online]. Available: http://www.jini.org/

[20] C. Bettstetter, G. Resta, and P. Santi, "The node distribution of the random waypoint mobility model for wireless ad hoc networks," *Mobile Computing, IEEE Transactions on*, vol. 2, no. 3, pp. 257–269, 2003. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1233531

[21] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing*, vol. 2, no. 5, pp. 483–502, 2002. [Online]. Available: http://dx.doi.org/10.1002/wcm.72

[22] I. Taylor, I. Downard, B. Adamson, and J. Macker, "AgentJ: Enabling Java NS-2 Simulations for Large Scale Distributed Multimedia Applications," in *Second International Conference on Distributed Frameworks for Multimedia DFMA 2006*, Penang, Malaysia, 14th to 17th May 2006.

[23] Naval Research Lab, "AgentJ: Java network simulations ins NS-2. An installation and user manual and source code." [Online]. Available: http://cs.itd.nrl.navy.mil/work/agentj/

[24] S. McCanne, S. Floyd, K. Fall, K. Varadhan *et al.*, "Network simulator ns-2," 2000.